REMARKS

Claims 4, 5, 7, 8, 11, 12, 15, 16, 19, 20, 21, 25, 41, 43, 45, 48, 49, 51, 52, and 56-62 are pending in the application.

Claims 7, 12, and 41 are amended to specify that the invention employs a plurality of annotators or means for annotating. Claims 7, 12, 16, and 41 are amended to change "tree-based functionality" to the more accurate --tree traversal functionality based on a language that can navigate XML representations of text--. Claims 7, 12, 16, 20, and 41 are amended to recite that the invention is implemented using a client-server hardware architecture. As noted in paragraph 000223 of the application, this "architecture allows for true client-server interaction, it also allows for a reasonable migration to a single-machine solution, in which both the client and server parts are installed on the same workstation." Claim 43 is amended to correct a typographical error.

These changes are believed not to introduce new matter, and entry of the Amendment is respectfully requested.

Based on the above Amendment and the following Remarks, Applicant respectfully requests that the Examiner reconsider all outstanding objections and rejections, and withdraw them.

Objections to the Specification

In section 6 of the Office Action, the amendment to the specification filed August 26, 2008 was objected to under 35 U.S.C. § 132(a) on the basis that it introduced new matter into the disclosure, specifically, the addition to paragraphs [00099] and [000115], that "FEX annotations are captured in a single view of the document expressed as inline XML" and in paragraphs

[000175] and [000186], the addition "in a single view of the annotated document." This objection is respectfully traversed on the ground that, as demonstrated by the Declaration Under 37 CFR § 1.132 of inventor Mark Wasson submitted herewith, a person of ordinary skill in the art would recognize this feature to be an inherent characteristic of and necessarily present in the invention from reading the description of the invention as described in the application as originally filed. See MPEP 2163.07(a) ("By disclosing in a patent application a device that inherently performs a function or has a property, operates according to a theory or has an advantage, a patent application necessarily discloses that function, theory or advantage, even though it says nothing explicit concerning it. The application may later be amended to recite the function, theory or advantage without introducing prohibited new matter. *In re Reynolds*, 443 F.2d 384, 170 USPQ 94 (CCPA 1971); *In re Smythe*, 480 F. 2d 1376, 178 USPQ 279 (CCPA 1973)").

Rejection under 35 U.S.C. § 112, ¶ 1

In section 8 of the Office Action, claims 57-59 were rejected under section 112, first paragraph, as failing to comply with the written description requirement, on the basis that the limitation of "annotating the text represents the annotated text as a single view of the document expressed as inline XML" not defined in the Specification as originally filed. This rejection is believed to be overcome by the accompanying Declaration Under 37 CFR § 1.132 of inventor Mark Wasson, for the reasons stated above with respect to the objections to the specification.

Rejections under 35 U.S.C. § 101

In section 9 of the Office Action, claims 7, 12, 20 were rejected under section 101 as being directed to a software embodiment and not to a hardware embodiment. In section 10, claims 4, 5, 8, 11, 15, 21, 25, 56-61 were rejected under section 101 as being dependent upon a rejected base claim.

In section 11, claims 16 and 19 were rejected under 101 on the basis that the claims do not appear to be tied to any statutory category, i.e. methods, machine, composition of matter, or article of manufacture.

In section 12, claims 41, 43, 45, 48, 49, 51, 52, 59, and 62 were rejected under section 101 as being drawn to non-statutory subject matter, on the ground that although they fall within the statutory category of process, the process 1) is not tied to another statutory class or 2) does not transform underlying subject matter to a different state or thing.

The rejections under section 101 are believed to be overcome by the foregoing amendments to independent claims 7, 12, 16, 20, and 41 tying the invention to the client-server hardware architecture described in the application.

Rejections under 35 U.S.C. § 103

In section 14 of the Office Action, claims 4, 7, 8, 12, 15, 41, 43, 48, 49, 52, and 56-62 were rejected under section 103 as being unpatentable over Cunningham et al. in view of Simov et al. and Krauthammer et al.

In section 15 of the Office Action, claims 5, 20, 21, 25, 45 (a typographical error in the statement of the rejection reads "41" instead of "45") were rejected under section 103 as being

unpatentable over Cunningham et al. in view of Simov et al. and Krauthammer et al., and further in view of Cunningham et al. (2).

In section 16 of the Office Action, claims 16 and 19 were rejected under section 103 as being unpatentable over Cunningham et al. in view of Simov et al.

In section 17 of the Office Action, claims 11 and 51 were rejected under section 103 as being unpatentable over Cunningham et al. in view of Simov et al. and Krauthammer et al., further in view of Marcus et al.

To the extent the Examiner may consider these rejections to be applicable to the claims as amended, they are respectfully traversed as being based on combinations of references that do not teach or suggest the claimed invention.

Tree traversal functionality

Part of the present invention is directed to the combination of regular expression functionality and tree-traversal functionality (using a language, such as XPath, that can navigate XML representations of text[1]) in a pattern recognition language used to match annotated text represented in XML. Independent claims 7, 12, 16, and 41 all recite that the text pattern recognition rules use "regular expression-based functionality, tree traversal functionality based on a language that can navigate XML representations of text, and user-defined matching functions."

The present invention combines regular expression-based pattern recognition functionality with a tree traversal-based pattern recognition functionality based on a language (such as XPath) that can navigate XML representations of text (which involves identifying two or more possibly non-contiguous (i.e., tree-structured) constituents. It is this integration of

[1] Paragraph [00079] of the application as filed defines XPath as "a language created to similarly navigate XML representations of texts." This definition is consistent with definitions found in computer science dictionaries.

regular expressions and tree traversal functionality that pulls a language like XPath into fact extraction. For natural language text processing purposes, tree traversal complements the left-right progression of regular expression rules by letting the rule-writer exploit syntactic relationships and not simply sequential relationships.

The Office Action asserts that it would have been obvious to combine regular expression capabilities and tree traversal capabilities, using XPath for the tree traversal work, in a pattern recognition language, based on the teachings of Cunningham et al. in view of Simov et al., and (with respect to some of the claims) further in view of Krauthammer et al., Krauthammer et al. and Cunningham et al. (2), or Krauthammer et al. and Marcus et al.

In general, the Office Action relies on publications associated with the GATE platform (Cunningham et al. and Cunningham (2)) for providing general teachings of the text annotation and information extraction problem space. The Office Action repeatedly notes GATE's general fact extraction capabilities.

Much of Cunningham et al. describes a range of characteristics common to a lot of text processing applications. Cunningham (2) is a published short research paper that highlights some features of the GATE architecture, whereas Cunningham et al. (the GATE user guide) is a more detailed source of information about the same technology. There is a considerable amount of literature on GATE, of which Cunningham et al. and Cunningham (2) are only two examples. According to inventor Mark Wasson, although the literature shows the evolution of the technology over time, overall the descriptions are quite consistent.

The Office Action concedes that Cunningham et al. do not teach annotation of tree based attributes and user-defined matching functions and tree-based functionality; and relies on Simov et al. as supplying these teachings, stating that "[i]t would have been obvious ... to have modified

the fact extraction tool as taught by Cunningham with the tree-based attribute, tree based functionality and user defined function as taught by Simov for the purpose of extracting certain information exceeding conditions in order to present the user with accurate information," and citing Section 4.5 of the Simov et al. article.

It is respectfully submitted that the extraction described in Section 4.5 of the Simov et al. is not fact extraction; and that Simov et al. teach using XPath only as a basic element selection function.

Regarding pattern recognition and XPath (or any other language that can navigate XML representations of text), it is first noted that the Simov et al. paper describes a tool that is primarily for creating, maintaining, and editing a linguistics knowledge base calls the BulTreeBank.

In summary, Simov et al.'s tool is not a fact extraction system, although it has some components in common with a fact extraction system, in particular some regular expression pattern recognition capabilities that apply to linguistics units, XML elements and their features. Simov et al.:

- Use XPath as a means to access individual elements in an XML-based representation of text. Once an individual element is accessed, it may be extracted, sorted, edited, its attributes may be retrieved, etc.
- Do not teach or suggest using XPath for tree traversal, and particularly for parse tree traversal (e.g., go from a subject to its verb, subject to object, etc.)
- Do not teach or suggest integrating tree traversal pattern recognition functionality with their regular expressions-based pattern recognition functionality, whether through XPath or through other means.

Using XPath to identify a specific node is a basic use of XPath – it is, after all, one of the things XPath was created to do. The Simov et al. article does indeed teach using XPath to define

conditions under which extraction may take place, but that task is not part of the pattern recognition itself. Simov et al. do not use it for pattern recognition purposes, and the XPath functionality is not integrated with regular expression pattern recognition. XPath-based pattern recognition functionality, such as tree traversal functionality, is not part of the regular grammar-based pattern recognition functionality described in the Simov et al. article. Instead, conditions may be used to determine when regular expression-based pattern recognition may apply.

Where Simov et al. do combine XPath with their regular expression-based pattern recognition is in using XPath to retrieve attributes from some element for use by the regular expressions. This operation impacts the attributes to which the regular expressions apply, but it does not introduce any new pattern recognition functionality to their regular expressions-based approach, whether tree traversal or any other pattern recognition functionality. It is still just a regular expression-based pattern recognition language.

Because of the way that Simov et al describe their XPath use – they mention some rules, and in one use the individual identified elements are extracted – the Office Action seems to equate this XPath use with fact extraction. However, as explained above, the use of XPath in Simov et al. is a very basic element identification task, identifying individual elements based on known attributes. It would have to be integrated with other pattern recognition tools, like regular expressions, to be able to exploit context, which is necessary for identifies roles, relationships and new attributes, basic fact extraction tasks.

Because Simov et al. describe using XPath in only a very basic way, and because they have not attempted to integrate it with sophisticated pattern recognition functionality like regular expressions, their approach does not provide any insight for someone skilled in the art to

incorporate XPath (or any other language that can navigate XML representations of text) for tree traversal or other interesting functionality in a pattern recognition language.

The Simov et al. article discusses XPath in several places surrounding its discussion of pattern recognition in XML-based content. In a number of ways, the teachings of the Simov et al. article are similar to those of Tokuda and Feldman, which were cited in the previous Office Action. XPATH is a powerful tool for processing XML-based content, so it is not unreasonable for Simov et al. to use it for a handful of tasks. However, pattern recognition is not one of those tasks. Using XPath to identify individual elements only demonstrates a fundamental capability of XPath. It does not provide any insight into how XPath (or any other language capable of navigating XML representations of text) might be integrated into a fact extraction pattern recognition language.

The Simov et al. article does not mention integrating XPath tree traversal capabilities into its regular expression-based pattern recognition process. It also does not describe the task of exploiting syntactic relationships in pattern recognition, in spite of the numerous references to syntax.

The most relevant part of the Simov et al. article is Section 4.3. The approach described there by Simov et al. uses regular expression grammars as the basis for its fact extraction-like pattern recognition. XPATH contributes here, but only as a means for providing XML element values to the pattern recognition rules in the regular grammar. Retrieving XML element values is a standard use of XPATH, but one that only provides attributes that the regular expressions may exploit during pattern recognition. It does not provide pattern recognition functionality itself, including any tree traversal pattern recognition functionality. Paragraph 2 under Section 4.5 also separates the XPath engine from the regular grammar engine.

The Simov et al. article also describes the use of XPath to support other functionality as well. Notable references to XPath uses include the following:

- Section 4.1, 1st paragraph, lines 7-8 ("we implemented an XPath language engine for navigation in documents"): This includes XML element value retrieval. Regular expression pattern recognition rules may apply to those values, but retrieving those values is not pattern recognition.
- Section 4.1, last paragraph, near end ("Some of these editing operations..."): The editing tools help their editors create and maintain the Bulgarian Treebank. However, editing operations are not pattern recognition processes.
- Section 4.4., all ("condition is stated as an XPath expression"): Constraints provide a means that guide what technologies and editorial processes may apply to what parts of some document. Although they may be used to determine where pattern recognition processes may apply, the constraints in and of themselves are not pattern recognition rules.
- Section 4.5., all ("condition is stated as an XPath expression"): XPath expressions are used to identify individual elements for some purpose, such as deleting or to support sorting. This is standard XPath functionality and it is not integrated into the regular expression pattern recognition capabilities. Editing, Sorting and Concordance functionality also is not fact extraction. However, this is also where they "extract" elements, which creates some of the confusion over fact extraction.

In sum, the Simov et al. article describes using XPath in standard ways to support some of the functionality that it provides with the Bulgarian Treebank, but the Simov et al. article does not describe using it for pattern recognition functionality that complements regular expression functionality in a pattern recognition language.

Simov et al. do not so much as hint at combining regular expression-based pattern recognition functionality with a tree traversal-based pattern recognition functionality based on a language (such as XPath) that can navigate XML representations of text, as recited in

independent claims 7, 12, 16, and 41, which is what pulls a language like XPath into fact extraction.

Using established, basic XPath functionality to perform one of its defined basic functions – individual element matching – not only is not fact extraction, but such use also does not provide any insight into how one might incorporate it into a fact extraction pattern recognition language for use in any type of sophisticated pattern recognition.

User-defined matching functions

With respect to the "user-defined matching functions," as recited in claims 7, 12, 16, and 41, the "functions" taught by the Simov et al. article are actually conditions, which are not part of a fact extraction process. Fact extraction-like pattern matching is accomplished by Simov et al. only through the use of regular expressions.

Resolving conflicting annotation boundaries

Another part of the present invention is directed to resolving conflicting annotations that have been marked up in some text using XML, because XML has a well-formedness requirement; that is, crossed annotation bracket pairs are not permitted in well-formed XML. Independent claims 7, 20, and 41 all recite "resolving conflicting annotation boundaries in the annotated text to produce well-formed XML." The Office Action cites the Krauthammer et al. paper with respect to the limitations related to the uncrossing process (i.e., resolving conflicting annotation boundaries).

The uncrossing limitations of claims 7, 12, 20, and 41 must be considered in the context of the accompanying limitations defining the annotators or means for annotating. Claims 7, 12, 20, and 41 also recite a plurality of independent annotators executed by the client-server hardware architecture for annotating the text with token attributes, constituent attributes, links,

and tree-based attributes, using XML as a basis for representing the annotated text, wherein each of the annotators has at least one specific annotation function. In other words, the present invention resolves conflicting annotation boundaries resulting from annotations produced by *multiple, independent annotators*.

When a system supports multiple annotators, there is usually a means provided for specifying the order that those annotators should apply. A text tokenizer will segment an input character stream into tokens such as words, spaces and punctuation symbols. A named entity recognizer might determine which sequences of tokens are people names. Fact extraction may apply to peoples' names and surrounding tokens to determine which people are attorneys and which are litigants. The order is often (but not always) important – named entity recognition expects to apply to tokens, not characters; fact extraction won't find attorneys if people names haven't been recognized. GATE, the present invention, and other extraction and mark up tool kits have a means for users to put their annotators into some order. Similarly, as Lexis and West put their legal content online, that content goes through a number of automated and/or editor steps in a specific order. However, none of the claims is directed to a process for ordering text annotators.

Further, whereas the order of the annotators may impact where and how they apply and thus their results, it does not necessarily have anything to do with XPath or similar languages, tree traversal, or overlap.

Annotators of course can annotate text with tree-like features to be traversed and they can create overlapping annotations, but that is separate from the ordering problem itself. In addition, ordering annotators implies that there are two or more annotators to apply to the text.

The Krauthammer et al. article addresses the problem of resolving annotation boundaries, but only for a single specific semantic components annotator that requires real world knowledge about the domain. It does not address other types of annotators or situations where the conflicts arise from applying multiple independent grammars.

Further, Krauthammer et al.'s approach is limited to a semantic concept tagger. While Cunningham (2) does describe the existence of a semantic tagger, it is a different one than the one Krauthammer uses, and the GATE literature (Cunningham et al. and Cunningham (2)) does not suggest that their own semantic tagger introduces overlap errors.

In the Krauthammer et al. article, the authors attempt to apply a single semantic concept annotation to the text. Some human language phenomena do not lend themselves to well-formed XML; the semantic scope of negation is one example on which the Krauthammer et al. article focuses, e.g., the text "no murmur or gallop" includes the semantic concepts "no murmur" and "no gallop." The Krauthammer et al. article does appear to work for the problem space they define, the semantic concept tagging of some text.

To summarize, the teachings of the Krauthammer et al. article suffer from a number of deficiencies with respect to the claimed invention, including the following:

- The approach taken by Krauthammer et al. is limited to the problems that arise when *a single specific annotator* introduces potential conflicts in the XML markup.
- In contrast, the present invention addresses the problem where two or more independent, potentially unknown annotators produce conflicting annotations. It is easier to propose a solution that maintains the integrity of a single known annotator than multiple, potentially unknown annotators.
- By only focusing on a semantic concept tagger, Krauthammer et al.'s solution is specifically tied to semantic tagging. For example, for their "erosions and bleeding in the antrum and fundus" example, their approach requires semantic world knowledge to know that erosions and bleeding take place on parts of the body AND that the

antrum and fundus are indeed parts of the body. Without access to such world knowledge properly applied to the data, the Krauthammer et al. approach fails, even on other semantic tagging tasks.

- In contrast, the present invention applies to any type of annotation results, many of which do not have anything comparable to the world knowledge on which Krauthammer et al. rely. Thus, the present invention does not rely on any sort of external knowledge to guide its uncrossing process.

Nowhere do Krauthammer et al. suggest that their approach generalizes across different types of annotators or in situations where two or more annotators are in conflict, areas that the present invention specifically addresses in claims 7, 12, 20, and 41.

Naming and defining a fragment of a pattern

With respect to claim 15, the Office Action characterizes the Cunningham et al. article as teaching "editing function to name ... and define a fragment of a pattern." In the art to which the present invention relates, editing functions are not fact extraction. Editing is what may be done with text *after* it is identified; that is, it is one possible post-extraction task. The authors of the Cunningham et al. article are simply describing a system they put in place to develop and maintain their Bulgarian Treebank, not a fact extraction system. Maintaining a Treebank requires editorial tools. Fact extraction does not.

Recognition of non-contiguous constituent attributes

With respect to claim 11, the Office Action cites Marcus et al. as teaching the identification of non-contiguous attributes. It is noted that well-formed XML, such as results in the present invention from resolving conflicting annotation boundaries (as recited in claim 12, from which claim 11 depends), has no problem encoding "non-contiguous" nodes. It is *overlapping* nodes (i.e., conflicting annotation boundaries) that are at issue, and that are resolved in the present invention by resolving the conflicting annotation boundaries.

Non-contiguous nodes are nodes that are not adjacent to one another. Because something separates them, their annotations do not overlap, and thus it is possible to address non-contiguous nodes with well-formed XML. If two non-contiguous nodes are related in some way, regular expression-based pattern recognition may not be appropriate for matching them, because the pattern recognition will *also* have to account for everything that might possibly be found between them. It is in this case where functionality like tree traversal comes in handy, because tree traversal can exploit tree syntax to bypass whatever appears between the two non-contiguous nodes.

Both the present and the previous Office Action associate the non-contiguous nodes problem to the overlapping problem, with the citation of Broder in the previous Office Action and Krauthammer in the present Office Action.

Because there is no inherent overlapping problem in non-contiguous nodes, the references to Marcus in the context of the Krauthammer et al. article are still puzzling, even if the Krauthammer et al. article taught a general purpose algorithm not limited to their own semantic concept tagger annotations. The Marcus paper refers to these as "discontinuous constituents" in its focus on syntactic parse trees, leading to "non-contiguous structure." The Marcus paper uses "simple notational devices" to indicate whether two discontinuous constituents are related within the syntactic tree, but it does all of this within a well-formed bracketed (parentheses, actually) structure comparable to how XML would do it

Furthermore, the model described in the Krauthammer et al. article can only address overlapping nodes for a single semantic concept tagger because it critically relies on extensive domain knowledge. In contrast, the present invention does not rely on knowledge bases, nor is it limited to a single type of annotator.

In view of the foregoing, it is respectfully submitted that the invention as recited in independent claims 7, 12, 16, 20, and 41, and the claims depending therefrom, is patentable over the cited prior art; and that the rejections should be withdrawn.

Conclusion

All objections and rejections have been complied with, properly traversed, or rendered moot. Thus, it now appears that the application is in condition for allowance. Should any questions arise, the Examiner is invited to call the undersigned representative so that this case may receive an early Notice of Allowance.

Favorable consideration and allowance are earnestly solicited.

Respectfully submitted,

JACOBSON HOLMAN PLLC

Date: May 14, 2009

By: _____

Allen S. Melser
Registration No. 27,215

Customer No. 00,136
400 Seventh Street, N.W.
Washington, D.C. 20004
(202) 638-6666

Enclosures: **Declaration Under 37 CFR § 1.132 of Mark Wasson**
Petition For Extension Of Time
Credit card payment form